(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 967 803 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
29.12.1999 Bulletin 1999/52

(51) Int. Cl.[6]: **H04N 7/24**

(21) Application number: **99111935.5**

(22) Date of filing: **23.06.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **24.06.1998 US 90532**
17.07.1998 US 118467

(71) Applicant: **NEC CORPORATION**
Tokyo (JP)

(72) Inventors:
- **Cox, Ingemar J.**
  Princeton, New Jersey 08540 (US)
- **Miller, Matthew L.**
  Princeton, New Jersey 08540 (US)
- **Oami, Ryoma**
  Minato-ku, Tokyo (US)

(74) Representative:
VOSSIUS & PARTNER
Siebertstrasse 4
81675 München (DE)

(54) Robust digital watermarking

(57) A watermarking procedure that is applicable to images, audio, video and multimedia data to be watermarked divides the data to be watermarked into a set of n×n blocks, such as the 8×8 blocks of MPEG. The same watermark signal can be distributed throughout the set of blocks in a large variety of ways. This allows the insertion algorithm to be changed without affecting the decoders. The decoding procedure first sums together the DCT coefficients of N sets of 8×8 blocks to form a set of N summed 8×8 blocks and then extracts the watermark from the summed block. Since the sum of the DCT blocks is equal to the DCT of the sum of the intensity blocks, efficient decoding can occur in both the spatial and frequency domains. The symmetric nature of the decoding process allows geometric distortions to be handled in the spatial domain and other signal distortions to be handled in the frequency domain. Moreover, insertion of a watermark signal into image data and the subsequent extraction of the watermark from watermarked image data which has been subject to distortion between the times of insertion and extraction involves the insertion of multiple watermarks designed to survive predefined distortions of the image data, such as pan-scan or letterbox mode transformations. Alternatively, a registration pattern in the image data, after the image data containing the registration pattern is subject to an unknown distortion, is used to compensate for distortion of the watermarked image data.

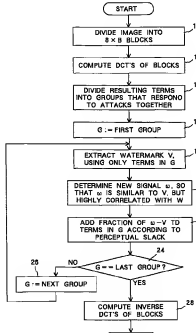

FIG. 1

EP 0 967 803 A2

Description

[0001] The present invention relates to digital watermarking of data including image, video and multimedia data. Specifically, the invention relates to insertion and detection or extraction of embedded signals for purposes of watermarking, in which the insertion and detection procedures are applied to sums of subregions of the data. When these subregions correspond to the 8×8 pixel blocks used for MPEG (moving picture experts group) and JPEG (joint photograph experts group) compression and decompression, the watermarking procedure can be tightly coupled with these compression algorithms to achieve very significant savings in computation. The invention also relates to the insertion and detection of embedded signals for the purposes of watermarking, in which the watermarked data might have undergone distortion between the times of insertion and detection of the watermark.

2. Description of the Prior Art:

[0002] The proliferation of digitized media such as image, video and multimedia is creating a need for a security system that facilitates the identification of the source of the material.

[0003] Content providers, i.e. owners of works in digital data form, have a need to embed signals into video/image/multimedia data, which can subsequently be detected by software, and/or hardware devices for purposes of authentication of copyright ownership, and copy control and management.

[0004] For example, a coded signal might be inserted in data to indicate that the data should not be copied. The embedded signal should preserve the image fidelity (image quality), be robust to common signal transformations and resistant to tampering. In addition, consideration must be given to the data rate that can be provided by the system, though current requirements are relatively low --a few bits per frame.

[0005] In U.S. Patent Application Serial No.08/534,894, filed September 28, 1995, entitled "Secure Spread Spectrum Watermarking for Multimedia Data", which is incorporated herein by reference, there was proposed a spread spectrum watermarking method which embedded a watermark signal into perceptually significant regions of an image for the purposes of identifying the content owner and/or possessor. A strength of this approach is that the watermark is very difficult to remove. In fact, this method only allows the watermark to be read if the original image or data is available for comparison. This is because the original spectrum of the watermark is shaped to that of the image through a non-linear multiplicative procedure, and this spectral shaping must be removed prior to detection by matched filtering. In addition, the watermark is usually inserted into the N largest spectral coefficients, the ranking of which is not preserved after watermarking. This method does not allow software and hardware devices to directly read embedded signals without access to the original unwatermarked material.

[0006] In an article by Cox et al., entitled "Secured Spectrum Watermarking for Multimedia", available through the Internet at http://www.neci.nj.nec.com/tr/index.html (Technical Report No. 95-10) spread spectrum watermarking is described which embeds a pseudo-random noise sequence into the digital data for watermarking purposes.

[0007] The above prior art watermark extraction methodology requires the original image spectrum be subtracted from the watermark image spectrum. This restricts the use of the method when there is no original image or original image spectrum available to the decoder. One application where this presents a significant difficulty is for third party device providers desiring to read embedded information for operation or denying operation of such a device.

[0008] In U.S. Patent No. 5,319,735 by R. D. Preuss et al. entitled "Embedded Signaling" digital information is encoded to produce a sequence of code symbols. The sequence of code symbols is embedded in an audio signal by generating a corresponding sequence of spread spectrum code signals representing the sequence of code symbols. The frequency components of the code signal being essentially confined to a preselected signaling band lying within the bandwidth of the audio signal and successive segments of the code signal corresponds to successive code symbols in the sequence. The audio signal is continuously frequency analyzed over a frequency band encompassing the signaling band and the code signal is dynamically filtered as a function of the analysis to provide a modified code signal with frequency component levels which are, at each time instant, essentially a preselected proportion of the levels of the audio signal frequency components in corresponding frequency ranges. The modified code signal and the audio signal are combined to provide a composite audio signal in which the digital information is embedded. This component audio signal is then recorded on a recording medium or is otherwise subjected to a transmission channel. Two key elements of this process are the spectral shaping and spectral equalization that occur at the insertion and extraction stages, respectively, thereby allowing the embedded signal to be extracted without access to the unwatermarked original data.

[0009] In U.S. Patent No. 5,848,155 by Cox entitled "A Spread Spectrum Watermark for Embedded Signaling" and incorporated herein by reference, there is described a method for extracting a watermark of embedded data from watermarked images or video without using an original or unwatermarked version of the data.

[0010] This method of watermarking an image or image data for embedded signaling requires that the DCT (discrete cosine transform) and its inverse (IDCT) of the entire image be computed. There are fast algorithms for computing the DCT in N log N time, where N is the number of pixels in the image. However, for N=512×512, the computational require-

ment is still high, particularly if the encoding and extracting processes must occur at video rates, i.e. 30 frames per second. This method requires approximately 30 times the computation needed for MPEG-II decompression.

[0011] One possible way to achieve real-time video watermarking is to only watermark every *N*-th frame. However, content owners wish to protect each and every video frame. Moreover, if it is known which frames contain embedded signals, it is simple to remove those frames with no noticeable degradation in the video signal.

[0012] An alternative option is to insert the watermark into n×n blocks of the image (subimages) where n<<N. If the block size is chosen to be 8×8, i.e. the same size as that used for MPEG image compression, then it is possible to tightly couple the watermark insertion and extraction procedures to those of the MPEG compression and decompression algorithms. Considerable computational saving can then be achieved since the most expensive computations relate to the calculation of the DCT and its inverse and these steps are already computed as part of the compression and decompression algorithm. The incremental cost of watermarking is then very small, typically less than five percent of the computational requirements associated with MPEG.

[0013] U.S. Patent Application Serial No.08/715,953, filed September 19, 1996, entitled "Watermarking of Image Data Using MPEG/JPEG Coefficients" which is incorporated herein by reference, advances this work by using MPEG/JPEG coefficients to encode the image data.

[0014] U.S. Patent Application Serial No.08/746,022, filed November 5, 1996, entitled "Digital Watermarking", which is incorporated herein by references, describes storing watermark information into subimages and extracting watermark information from subimages.

[0015] A review of watermarking is found in an article by Cox et al., entitled "A review of watermarking and the importance of perceptual modeling" in Proc. of EI'97, vol. 30-16, February 9-14, 1997.

[0016] There have been several proposals to watermark MPEG video or JPEG compressed still images. In all cases, each 8×8 DCT block is modified to contain the watermark or a portion thereof. Consequently, decoding of the watermark requires that each 8×8 block be individually analyzed to extract the watermark signal contained therein. The individual extracted signals may then be combined to form a composite watermark, which is then compared with known watermarks. Because each block must be analyzed individually, an uncompressed image must be converted back to the block-based DCT representation, which is computationally expensive. Thus, while the decoder may be computationally efficient in the DCT domain, extracting a watermark from the spatial domain is much more expensive.

[0017] To allow for computationally efficient detection of the watermark in both the spatial and DCT domains, a watermark may be inserted in the sum of all the 8×8 blocks in the DCT domain, or the sum of a subset of all the 8×8 blocks in the DCT domain. A major advantage of this approach is that if the image is only available in the spatial domain, then the summation can also be performed in the spatial domain to compute a small set of summed 8×8 blocks and only those blocks must then be transformed into the DCT domain. This is because the sum of the DCT blocks is equal to the DCT of the sum of the intensities. Thus, the computational cost of decoding in the DCT and spatial domains is approximately the same.

[0018] A second advantage of watermarking the sum of the DCT blocks is that there are an unlimited number of equivalent methods to apportion the watermark throughout the image. For example, if the watermark requires a change of Δi to the i-th coefficient of the summed DCT block, then, if there are M blocks in the image, $\Delta i/M$ can be added to each individual block, or block 1 can have Δi added to it and the remaining M-1 blocks left unaltered, ignoring for the moment issues of image fidelity. Because of this one to many mapping, it is possible to alter the insertion algorithm without changing the decoder. This is a very important characteristic, since in some watermarking applications, there may be many hardware decoders that are deployed, such that changing the decoder is impractical. However, improvements to the insertion algorithm can still result in improved detection using the approach described herein.

[0019] A third advantage of watermarking the sum of the DCT blocks is that watermark signals extracted from these sums have small variances, compared with the amount that they may be changed without causing fidelity problems. This means that, in many cases, it is possible to change an image so that the summed DCT blocks perfectly match the required watermark signal, even though the resulting image appears identical to the original.

[0020] Finally, it is well known that some problems, such as modeling the human visual system, are best performed in the frequency domain, where other problems such as geometric transformations are more conveniently dealt with in the spatial domain. Since the computational cost of decoding the watermark is now symmetric, it is possible to switch from spatial to frequency domains at will in order to correct for various signal transformations that may corrupt the watermark.

[0021] The present invention concerns a novel insertion method which employs a specific model of the human visual system which provides much better control over image fidelity. Tests have shown that it is possible to obtain large signals (more than 15 standard deviations from 0 correlation) with images that are indistinguishable from their respective original images.

[0022] The method handles robustness against various types of attacks in ways that are easy to relate to the specific type of attack.

[0023] The method is adaptable so that the model of the human visual system and the techniques used for handling

attacks can be changed later without having to change the detector. The result is that it is possible to continue improving watermarking, particularly DVD (digital video disk) watermarking, even after many detectors have been installed. This is analogous to the situation with MPEG video for which encoder technology can be improved without having to change existing decoders.

[0024] Use of the present insertion method allows a simple detection algorithm in either MPEG or decompressed domains.

[0025] The invention also concerns a novel detection method which is easy to implement, easy to analyze and has a low computational cost, whether the incoming video is MPEG compressed or uncompressed.

[0026] The present invention also concerns a novel insertion method that hides multiple patterns in the data. These patterns fall into two categories: 1) registration patterns used during detection to compensate for translational shifts, and 2) watermark patterns that encode the information content of the watermark.

[0027] A principal object of the present invention is the provision of a digital watermark insertion method which allows detection of watermarks after the watermarked data is subjected to predefined scale changes, without modification to the watermark detector.

[0028] Another object of the invention is the provision of a watermark detection method that is computationally inexpensive in either the MPEG or decompressed domains.

[0029] A still other object of the invention is the provision of a digital watermarking method that withstands attacks without having to change a detector.

[0030] Further and still other objects of the invention will become more clearly apparent when the following description is read in conjunction with the accompanying drawings.

Brief Description of the Drawings

[0031]

Figure 1 is a general data flow diagram of a method of inserting a watermark into media data;
Figure 2 is a schematic diagram of an MPEG-2 encoder;
Figure 3 is a schematic diagram of a modified MPEG-2 encoder for reducing degradation of a watermark in watermarked data;
Figure 4 is a schematic diagram of an alternative modified MPEG-2 encoder for reducing degradation of a watermark in watermarked data;
Figure 5 is a flow chart of the process performed in the watermark correction device in Figures 3 and 4;
Figure 6 is a flow chart of the process performed in step 506 of Figure 5;
Figure 7 is a flow chart of an alternative process performed in step 506 of Figure 5;
Figure 8 is a flow diagram of a method of extracting a watermark from MPEG media data;
Figure 9 is a flow diagram of a method of extracting a watermark from uncompressed media;
Figure 10 is a flow diagram of a method of detecting a watermark from MPEG data, with registration; and
Figure 11 is a flow diagram of a method of detecting a watermark from uncompressed image data, with registration.

Detailed Description of the Preferred Embodiment

[0032] As used in the following description the terms image and image data will be understood to be equally applicable to video, image and multimedia data. The term "watermark" will be understood to include embedded data, symbols, images, instructions or any other identifying information.

[0033] In order to better understand the present invention, first a review of the basic watermarking method will be presented followed by additional descriptions of the improvements comprising the present invention.

[0034] First, we define some notations. Let a watermark to be embedded into an image be an N dimensional vector, denoted by W[1,...,N]. In the following text, the notation W[1,...,N] is used in the same manner as W[k] (k=1,...,N). Let V[1,...,N] denote a vector value extracted from an image, where the element V[k] corresponds to W[k]. Specifically, the value V[k] is a weighted sum of DCT coefficients given by

$$V[k]=D\_k[1]F\_k[1]+D\_k[2]F\_k[2]+...+D\_k[n\_k]F\_k[n\_k],$$

where D_k[i] (i=1,...,n_k) indicate members of the set of DCT coefficients used for calculating V[k], n_k indicates the number of members, and F_k[i] (i=1,...,n_k) are weighting coefficients related to a filter processing. The concept of F_k is that the DCT coefficients are weighted according to how much noise might be expected in each coefficient. To calculate V[1,...,N], n×n DCT coefficients are first calculated over a whole image. Then the coefficients are classified into N sets, each of which is related to each element of V[1,...,N]. The rule of classifying DCT coefficients is predetermined,

and the same rule is used in both inserting and detecting a watermark.

[0035] Before inserting a watermark into an image, a detection algorithm is applied to the image to find a watermark that is already present in the image. If the image does not contain a watermark, the extracted values, V[1,...,N] will be normally distributed random numbers that do not correlate any watermark W[1,...,N]. A watermark W[1,...,N] is inserted into an image by changing each of D_k[1,...,n_k] (k-i,...,N) slightly in order to make the extracted value V[1,..., N] highly correlate the watermark W[1,...,N]. Let the target value of V[1,...,N] be denoted by ω[1,...,N], that is, the values V[1,...,N] are changed to ω[1,...,N] by inserting the watermark W[1,...,N]. The target values ω[1,...,N] have high correlation with the watermark W[1,...,N] and they are determined as will be described below.

[0036] After the target values ω[1,...,N] are determined, the difference ω[k]-V[k] is distributed among the DCT coefficients D_k[1,...,n_k]. Then a watermark is inserted by adding the allocated difference value to the corresponding DCT coefficients D_k[1,...,n_k]. This change of DCT coefficients must be done in such a manner so as not to change the appearance of the image.

[0037] In distributing the difference ω[k]-V[k] , a characteristic of the human visual process is taken into account. The amount of change that does not cause a visible change in the image is different for each DCT coefficient D_k[i]. This amount depends on the human visual process which can be approximately simulated with a computational model. The amount of change is referred to as "slack". The slack is calculated for each DCT coefficient and it is used in distributing the difference ω[k]-V[k] among the DCT coefficients. Next we describe how to calculate the values of the slack using a model of the human visual process.

[0038] The preferred computational model of human visual sensitivity that is used in the present invention is found in an article by Andrew B. Watson, entitled "DCT Quantization Matrices Usually Optimized for Indirect Images" in SPIE, vol. 1913 (1993), pp. 202-216. This model was applied to watermarking in an article by Christine I. Podilchuk and Wenjun Zeng entitled "Digital Image Watermarking Using Visual Models", Proc. of EI'97, vol. 3016, February 9-14, 1997. The current invention differs from that of Podilchuk and Zeng in (i) not requiring the original unwatermarked image at the decoder and (ii) not extracting the watermark from the individual 8×8 blocks, but from the sum of a set of 8×8 blocks. Other computational models are also usable.

[0039] For each element of the image's block DCT, d[i,j], this model computes a value called the element's "slack", S[i,j], which indicates how much a particular d[i,j] value may be altered before such an alteration becomes visible. The value is computed in three steps. The first step models the contrast masking phenomenon of the human visual system and models the visual sensitivity at different frequencies and handles the difference between visual sensitivity to changes in different frequencies. The second step models the luminance masking phenomenon of the human visual system and handles the fact that the visual system is more sensitive to changes in dark regions than to changes in bright regions. The third step handles the fact that the sensitivity to changes depends in part on the percentage that the frequency is changing (i.e. a DCT term with a small value in it may only change a little, while one with a larger value may change more).

[0040] The perceptual model makes use of a matrix of values that indicate the relative sensitivity of the human visual system to the different terms of a spatial 8×8 DCT. The formulae for computing this matrix are available in an article by Albert J. Ahumada Jr. and Heidi A. Peterson entitled "Luminance-Model-Based DCT Quantization for Color Image Compression", in SPIE, vol. 1666, (1992) pp. 365-374.

[0041] After computing the slacks for all the 8×8 DCT's in the image, a slack can be assigned to each D_k[1,...,n_k]. Call these slacks S_k[1,...,n_k]. It is now possible to distribute the changes in the V's over all the D_k's with minimal visual impact. This is done according to the following formula:

$$D'\_k[l] = D\_k[l] + \frac{(\omega[k] - V[k]) * S\_k[l]}{\sum_{j=1}^{n\_k} S\_k[j] F\_k[j]}$$

where D'_k[1,...,n_k] are the modified 8×8 DCT coefficients, and ω[k] and V[k] are the k-th elements of ω and V, respectively. The effect of this formula is to distribute the desired change in a given element of the watermark vector (ω[k] - v[k]) over all the DCT coefficients that are summed at calculation of V[k] to produce that element, proportionately according to those DCT coefficients' slacks. To illustrate, consider two simple examples: 1) If all the slacks are 0 except for slack S_k[m], then the sum of all the values S_k[j]F_k[j] is equal to S_k[m]F_k[m], and only D_k[m] is changed. It is changed by the full value of (ω[k]-V[k]). 2) If all the slacks are equal and all the coefficients F_k[i]=1, then each D_k[i] is changed by the same amount.

[0042] After making these changes, convert all the 8×8 DCT's back into the spatial domain, and the result is a watermarked image. It is easy to show that the sum of all the D_k[i]F_k[i]'s for a given k will equal ω[k]. The process of making this is referred to as "inserting Omega into the image". The watermark extracted from the resulting image, if the image has not been attacked, will be exactly ω, not ω plus noise.

[0043] There are two important issues remaining to be discussed. First, how to decide on ω, and, second, how to make the watermark robust.

[0044] Previously, the equivalent of ω was computed as:

$$\omega = V + \alpha * W$$

where α is a small constant, and W is a zero-mean watermark signal. It is possible to use the same formula here, but it is too limiting to result in the strongest possible watermark using the present invention. In practicing the invention, it is often possible to insert an ω that has perfect correlation with the watermark, W, without causing any visible change in the image. The following formula is used:

$$\omega = \text{mean}(V) + \beta * (V - \text{mean}(V)) + \alpha * W$$

[0045] This result is a weighted sum of the watermark signal and the original, noise (image) signal. If β is set to 0, the result is an ω that perfectly correlates with W.

[0046] The signal to noise ratio (SNR) for an unattacked image will be:

$$\text{SNR} = \alpha * \text{std}(W)/\beta * \text{std}(V)$$

where std(X) is the standard deviation of X.

[0047] There are many ways to choose α and β based on optimizations to maximize different criteria such as fidelity or robustness.

[0048] At this point there is a complete method of inserting watermarks. The method contains explicit modeling of human vision, but it does not contain any explicit method of making the watermark robust. In fact, the method as described so far will try to put as much of the watermark as possible into the high frequencies since these frequencies have the largest slack, but this is a poor thing to do from the point of view of robustness.

[0049] To make the watermark robust against a given set of attacks or signal degradations, it is first necessary to consider how those attacks affect the various terms of the 8×8 DCT's in the image. Then, terms that are affected by attacks or signal degradations in similar ways are grouped together, and watermarked as if the group of terms were a separate image.

[0050] The following is a simple example. Suppose there is only concern about two possible attacks: cropping 24 columns of pixels from the left side of the image, or cropping 24 columns of pixels from the right side of the image. This results in three groups of DCT terms: those that come from the 3 left-most columns of 8×8 DCT's blocks, those that come from the 3 right-most columns, and those that come from the rest of the image. All the terms in each of these groups either survives or is destroyed by any given attack together. If each group is watermarked as though it were a separate image, then the watermark from at least one group will generally survive attack (assuming that the 24-column cropping attacks are the only attacks possible), and the watermark that is extracted will consist of the correct watermark, from that group, plus some noisy watermarks from groups that were damaged by the attack.

[0051] A more interesting example is low-pass and high-pass filtering attacks. It is possible to group all the low frequencies together into one group, and all the high frequencies into one or more other groups. If the predetermined rule for classifying DCT coefficients into the N sets is designed in such a way that each set has coefficients of many different frequencies, then the complete watermark can be inserted into each group. Then, if the high frequencies are removed, the watermark will still be detectable in the low frequencies, and vice-versa.

[0052] The more groups the terms are divided into, the more robust will be the watermark. There is a cost because it will become increasingly more difficult to distribute the changes without causing visible distortion.

[0053] It is important to note that the best balance can be achieved after the detectors are in wide use. It is possible to modify the insertion algorithm to make watermarks robust against a wide variety of attacks without having to change detection at all.

[0054] Presently only three groups are used. One group collects together most of the low frequencies. Each of the other two groups represents one higher frequency.

[0055] In the preferred method, a watermark is not placed in any of the higher frequencies. The reason is not that they are susceptible to attack (that is handled by the design of the filter used in detection). Rather, it is because watermarking the higher frequencies causes MPEG compression rates to go down substantially. An alternative solution might be to add other groups that contain higher frequencies.

[0056] In Figure 1, there is shown a flow diagram of the watermark insertion method. The digital image is divided into a collection of n×n blocks, preferably 8×8 blocks, in step 10. The discrete cosine transform (DCT) of each block is computed in a known manner in step 12. The DCT's are separated into groups that respond to different attacks in the same manner in step 14. A first group G is selected in step 16.

[0057] Next, extract a watermark V, using only the DCT terms in the group G in step 18. Determine a new signal (target value) ω selected such that ω is similar to V but is highly correlated with watermark W in step 20.

[0058] Add fractions of ω-V to the DCT terms in G according to perceptual slack in step 22. Decide whether group G is the last group in step 24. If not, select next group G in step 26, and extract watermark V using only terms of next group G in step 18 and continue procedures until the last group G is found in step 24. Next, compute the inverse DCTs of the blocks in step 28 resulting in a watermarked image.

[0059] Some alternative steps in the insertion method are possible. For example, the distribution of the difference between ω and V over DCT terms can be done stochastically to help deter tampering and reduce susceptibility to tampering.

[0060] Also, the groups of DCT terms for robustness purposes could be performed dynamically. For example, the insertion program could simulate various attacks on the image and determine the effect on the values of the DCT term. Then, the program would cause appropriate allocation of the terms into the groups similarly affected.

[0061] The inserter can be designed with a user-interface that allows the user to set two parameters: (1) the maximum perceptual difference (J) between the original image and the watermarked image and (2) the maximum allowable probability of missed detection after any of the predefined set of attacks. The algorithm would then insert watermarks into a large number of images automatically, according to the allowable perceptual change J and checking each one against simulations of the attacks. If an image fails to meet the specified robustness constraint (maximum allowable probability of missed detection), then the user would be notified so that a manual decision can be made to compensate or trade-off image fidelity for robustness.

[0062] In addition, the distribution of the difference between ω and V over DCT terms can be modified to explicitly compensate for MPEG quantization. Using the above watermark insertion method there may result a degraded watermark in the watermarked data. In order to enhance the watermark in the watermarked data after MPEG compression several techniques are possible.

[0063] Figure 2 is a schematic diagram of a typical MPEG-2 encoder. Figure 2 depicts elements which are indispensable to execute an MPEG-2 encoding of P pictures, or to perform a combined interframe prediction and DCT coding. Input images are provided as one input to subtractor 30. The other input to subtractor 30 is predicted image generated in frame memory 32. The predicted images are subtracted from the input images at subtractor 30. A discrete cosine transform (DCT) is performed at DCT calculator 34 on the output signal from subtractor 30. The DCT coefficients are quantized in quantizer 36. The outputs of the quantizer 36 are sent to a variable length encoder 38 where Huffman encoding is performed. The quantized DCT coefficients outputted from the quantizer 36 are also sent to an inverse quantizer 40 where they are de-quantized. Inverse DCT of the de-quantized DCT coefficients is performed in the inverse DCT calculator 42. The results are added at adder 44 to the predicted image outputted from the frame memory 32, and then an image which is expected to be the same as that acquired in a decoder is reconstructed. The reconstructed image is called "a locally decoded image." This locally decoded image is stored in the frame memory 32 to produce the predicted images.

[0064] Figure 3 is a schematic diagram of a modified MPEG-2 encoder for reducing degradation of a watermark in watermarked data. Before MPEG-2 encoding, DCT is performed on a input image at the DCT calculator 50 and watermark signals are added to the DCT coefficients at adder 52. The output DCT coefficients including watermark information is subject to inverse DCT in the inverse DCT calculator 54. The output of inverse DCT calculator 54 are images with a watermark. These watermarked images are sent to MPEG-2 encoder and MPEG-2 encoding is performed as described above. In addition, in this embodiment, watermark information is modified in order to be suited to MPEG-2 compression. DCT coefficients for the predicted images are calculated in DCT calculator 56. The quantization values outputted from quantizer 36 are de-quantized in inverse quantizer 58. The results of inverse quantization are added at adder 60 to the DCT coefficients outputted from DCT calculator 56. The results of addition correspond to the DCT coefficients for the decoded images which are expected to be generated in a decoder. These DCT coefficients are inputted into a watermark correction device 62. The watermark correction device 62 outputs watermark correction signals. At adder 64, the watermark correction signals from device 62 are added to the quantization values from quantizer 36. The output of adder 64 is used as the inputs to variable length encoder 38 and inverse quantizer 40.

[0065] Next, we describe the process performed in the watermark correction device 62. Let us introduce several new notations to explain the process. Let Dq_k[i] be the quantization value corresponding to D_k[i], that is, the quantization value of the i-th member of the k-th set for calculating the value V. Let Q_k[i] be the quantization step size used in obtaining Dq_k[i]. Let Dr_k[i] be the output value of adder 60 that is obtained by adding the inverse quantization value of Dq_k[i] calculated in inverse quantizer 58 to the corresponding DCT coefficient outputted from DCT calculator 56. Let Vr[1,...,N] be the value extracted from the output values of adder 60, Dr_k[i], in the same manner that the value V[1,...,N] is calculated in inserting a watermark. We assume that the target value o[1...,N] is also available in the watermark correction device 62.

[0066] Figure 5 shows the flow chart describing the process performed in the watermark correction device 62. First the index k of the watermark element is set to 1 at step 500. Next the value Vr[k] is calculated at step 502 by

$$Vr[k] = F\_k[1]Dr\_k[1] + ... + F\_k[n\_k]Dr\_k[n\_k],$$

where the weighting coefficients F_k[1,...,n_k] are the same value as those used in calculating V[k]. Then the absolute value of the difference between the value Vr[k] and the target value ω[k], and the sign of the difference are computed in step 504 by the following equations:

$$Dif = |Vr[k]-\omega[k]|$$

$$s = Sign(Vr[k]-\omega[k]),$$

where

$$Sign(x) = \begin{cases} 1 & (x \geq 0) \\ -1 & (x < 0) \end{cases}$$

This value Dif corresponds to the distortion of the watermark inserted at the adder 52 generated in the quantization process.

[0067] On the basis of the absolute value Dif and the sign S watermark correction signals are generated in step 506. The process in step 506 is described later. After the generation of the watermark correction signals for the DCT coefficients related to the *k*-th element of the watermark, index k is compared with N in step 508. If k > N, then the process is finished. If k ≤ N, the value k is increased by one in step 510, and the process goes back to step 502. The watermark correction process is thus performed, and the obtained watermark correction signals are finally outputted to adder 64.

[0068] Next, we describe the process performed in step 506 in Figure 5, using the flow chart in Figure 6. In step 506, the watermark correction signals for the DCT related to the k-th element of the watermark are generated. The array ΔDq_k[i] (i=1,...,n_k) are first all set to zero in step 520. Next, the value j is set to one in step 522. Then the index i of the DCT coefficients is found through a permuting function p(j) in step 523. The function p(j) returns the *j*-th value of a permutation obtained after the integers 1 to n_k are permuted. The simplest example is p(j)=j . Next, a value -s is stored in ΔDq_k[i], and the value Dif is decreased by Q_k[i]F_k[i] in step 524. This indicates that the quantization value Dq_k[i] is changed by -s by adding ΔDq_k[i] to Dq_k[i] at adder 64. The value s is 1 or -1, so the change in the quantization value is one. In other words, de-quantized value obtained in a inverse quantizer in a decoder is changed by -Q_k[i], one step size. The value Dif after the update is identical to the absolute value of the difference between ω[k] and Vr[k] calculated with the corrected quantization values Dq_k[i] + ΔDq_k[i] (i=1,...,n_k). After step 524, the values Dif and zero, and index j and n_k are compared in step 526. If Dif < 0 or j>n_k then this subroutine is finished. If the condition is not satisfied, the index j is increased by certain amount Δj in step 528, then the process returns to step 523.

[0069] Instead of the process shown in Figure 6, an alternative process shown in Figure 7 can be used as the process of step 506. In the process shown in Figure 7, a step 530, checking whether the quantization value Dq_k[i] equals zero or not, is added between step 523 and step 524. In this case, step 524 is performed only if the quantization value Dq_k[i] is a non-zero value. This allows a reduction in the increase of the number of bits caused by correcting watermark information, because changing a quantization value from zero to non-zero value generally results in a large increase in the number of bits.

[0070] Figure 4 is a schematic diagram of an alternative embodiment of a modified MPEG-2 encoder for reducing degradation of a watermark in watermarked data. In this embodiment, the basic concept is the same as the described in connection with Figure 3. The differences lie in the fact that subtraction of the predicted images from the input original images is performed in the DCT domain not in the spatial domain. For the predicted image outputted from the frame memory 32, DCT is performed in DCT calculator 70, and the results are subtracted from the watermarked DCT coefficients at subtractor 72. The results of subtraction are sent to the quantizer 36 and then the watermark correction is performed in the same manner as shown in Figure 3. The results outputted from subtractor 72 are the same as the results outputted from the DCT calculator 34 in Figure 3 because of the linearity of DCT. Therefore, the results obtained in the processes followed by quantization in quantizer 36 are the same as those in Figure 3. This embodiment results in a reduction in the number of DCT calculations.

[0071] The detection procedure to detect a watermark in an image will now be described.

[0072] If MPEG video is the input image data format, the following detection process determines whether watermark W is present, where W[1,...,N] is the watermark being tested for.

[0073] Decode the Huffman code, but do not compute the inverse DCT's, so that, for each frame (at least, each I-frame), there is an array of 8×8 DCTs.

[0074] Next perform the same summation of DCT coefficients that was performed during watermark insertion to

obtain the vector V. Compute the correlation coefficient C, between V and the watermark being tested for, W:

$$V' = V - \bar{V}$$

$$W' = W - \bar{W}$$

$$C = \frac{W' \bullet V'}{\sqrt{(W' \bullet W')(V' \bullet V')}}$$

[0075] Finally, convert C into a normalized Fisher Z statistic:

$$Z = \frac{\sqrt{N-3}}{2} \log \frac{1+C}{1-C}$$

where N is the length of the watermark.

[0076] The Z value indicates whether the watermark is present. A preferred threshold for Z is 4 (i.e. $Z \geq 4$ means the watermark is present), but other values may be used depending on the desired probabilities of false alarms and missed detections.

[0077] Figure 8 is a flow diagram of the detection method for MPEG video input described above. The input MPEG video is subject to a Huffman decoder and partial parser 80 where the output is a set of DCT for n×n, preferably 8×8, blocks of the video input.

[0078] The n×n DCTs are provided to watermark accumulator 82. Accumulator 82 has memory whose length is the watermark length N. DCT coefficients from the Huffman decoder and partial parser 80 are classified according to a predetermined rule and summed for extracting a watermark as mentioned before, and the results are accumulated in the memory. The extracted watermark is provided to comparator 84 where it is compared with possible watermarks in the image by calculating correlation coefficients between the extracted watermark and the possible watermarks as mentioned before. The possible watermarks are the universe of the watermarks that might have been inserted into the video data. The details of the accumulators and comparator are found, for instance, in U.S. Patent Application Serial No. 08/746,022.

[0079] The output of comparator 84 is the likelihood (normalized Fisher Z statistic) of the detected watermark being each of the possible watermarks. The most likely watermark is determined and is deemed the watermark in the image, or, if the detector does not exceed a predetermined threshold, then no watermark is present. Alternatively, if the incoming input data comprises an uncompressed image, an embedded watermark can be detected by applying the method above to DCT coefficients obtained by performing 8×8 DCT for the whole image. In this case, DCT have to be performed for each 8×8 block, but a skillfully designed rule for classifying DCT coefficients into N sets enables us to avoid performing DCT many times. Before explaining the method to reduce DCT calculation, let us define some notations.

[0080] Let h_m(i,j) (i=1,...,8, j=1,...,8, m=0,...,M-1) be a set of functions that map frequency indices of 8×8 DCT coefficients (i, j) onto the indices k of the element of a watermark, and M indicates the number of functions. So if k=h_m(i,j), then a DCT coefficient whose index is (i,j) is classified into the k-th set of DCT coefficients for calculating the value V[k]. We prepare M different functions h_m(i,j) (m=0,...,M-1). Which function is selected for a certain 8×8 block depends on the numbers r and c of the block where r and c indicate the row and column numbers of the block respectively. So the index of the functions, m, is first found according to the values r and c, then the index of the sets, k, is determined by h_m(i,j) for each DCT coefficient. Using these functions h_m(i,j) in classifying DCT coefficients is assumed in the remaining part of the detailed description.

[0081] In this case, we can reduce the number of DCT calculations in the following manner. First the sum of the blocks whose indices m are the same is computed for each m=0,...,M-1. Let this summed block be denoted by VB_m[i,j] (m=0,...,M-1). Then DCT is perfomred for the M summed blocks VB_m[i,j] (m=0,...,M-1). Finally, the DCT coefficients of the summed blocks are classified into N sets according to the value h_m(I,j), and added together within each set to obtain V[1,...,N]. The obtained results V[1,...,N] are the same as V[1,...,N] obtained with the method mentioned above because DCT is a linear transform, that is, the sum of DCT blocks equals the result of DCT for the sum of the blocks. If M is much less than the total number of blocks in an image, the number of DCT calculations is dramatically reduced. This method thus allows us to extract watermarks with small calculation cost.

[0082] Figure 9 shows a flow diagram of the detection method for uncompressed video input data as described above.

[0083] The uncompressed video image data is provided to n×n accumulators, preferably 8×8 accumulators 90. The memory requirement is n^2 times the number of the function h_m(i,j). For each index m, the blocks with the same index m are summed and the resultant M summed blocks are accumulated in the memory.

[0084] The output is the summed signal of each of the n×n blocks. The output is subject to a DCT transform 92. The number of transformations is proportional to the number of the functions h_m(i,j). The result is a group of n×n DCTs which is classified into N sets according to the functions h_m(i,j) and summed for extracting a watermark as mentioned before. The obtained watermark is provided to watermark accumulator 94 and accumulated. The memory requirement for accumulator 94 is proportional to the watermark length. The extracted watermark is provided as input to comparator 96. The other inputs to comparator 96 are the possible watermarks that may have been inserted into the input image data. The comparator computes a likelihood (normalized Fisher Z statistic) of each possible watermark having been inserted into the image data. The most likely watermark is determined and is deemed the watermark in the image.

[0085] A limitation of block based DCT methods is their sensitivity to spatial shifts of the image. For example, if the image is shifted two pixels to the right, then the DCT coefficients change significantly, so that the watermark cannot be detected. Furthermore, general distortions, such as scaling and rotation, also make the watermark undetectable.

[0086] To solve these problems, the above insertion and extraction methods may be modified in two ways. The first possible modification is to insert multiple watermarks designed to survive predefined distortions of the video. The second modification is to arrange that translations can be compensated for without performing the summation more than once. Optionally, this second modification may be further modified to insert registration patterns, one for each of the multiple watermarks, which can be used by a modified watermark detector to compensate for arbitrary translations of the video.

[0087] When detecting watermarks inserted using the above method, it is necessary to divide the image into the same grid of n×n blocks as was used during insertion. If the image has been translated since watermark insertion, then determining the correct grid becomes difficult. In many applications, this is a serious problem, since certain, specific transformations can be expected. For example, it can be expected that video on a DVD disk might be modified to fit on a standard television screen by conversion to either "panscan" or "letterbox" mode. In "panscan" mode, the horizontal image resolution is increased, and the image is cropped at a predetermined offset, so that the resulting image will be correct when viewed on a 3×4-aspect-ratio television screen. In "letterbox" mode, the image is scaled vertically, and black is added at the top and bottom, so that the whole image will fit correctly on a 3×4-aspect-ratio screen. Since these two geometric transformations are more likely than any other, it is reasonable to prepare for them specifically.

[0088] The problem of the predetermined scaling or transforming of watermarked video is solved in the present invention by inserting an additional watermark for each of the likely transformations. Each of these watermarks is designed so that, when the image has undergone the corresponding known transformation, the grid of n×n blocks used during insertion will align with a predetermined grid used during detection. Thus, if the image has undergone no transformation, then the detection grid will align with the normal mode watermark, and the normal mode watermark will be detected. If the image has undergone "panscan" transformation, then the same detection grid will align with the "panscan" watermark, and the "panscan" watermark will be detected, and so forth for "letterbox" scan or any other predefined transformation.

[0089] The procedure for inserting a watermark that is to be detected after a specific transformation comprises the following steps:

1. Make a copy, I_T, of the image being watermarked, I, and apply the transformation to be compensated for the image. For example, I_T might be a copy of I that has been transformed into "letterbox" mode by vertical shrinking of I.
2. Create a watermarked version of the transformed image, I_T', according to the general watermarking method described above.
3. Let $W_T = I_T' - I_T$ be the spatial pattern that was added to I_T when it was watermarked.
4. Perform the inverse transformation on W_T to yield the corresponding watermark pattern, W, for the untransformed image. For example, if the transformation to be compensated for was "letterbox" mode, then W would be obtained by vertically expanding W_T.
5. Let I' = I + W be the image with a watermark added for the given transformation.

[0090] When the transformation is applied to the watermarked image I', the result will be approximately I_T', and the watermark will be detected by the same procedure designed to detect a normal watermark.

[0091] This process can only be used for a small number of transformations, as each additional watermark causes additional degradation of the image, and reduces the detectability of other watermarks in the image. However, tests have shown that three watermarks --two for transformed images and one for the untransformed image--result in acceptable fidelity and good detectability. Alternatively, for video, each watermark can be inserted in a time-multiplexed manner.

[0092] In cases where the transformations that an image will undergo are not predefined or predetermined, or where there are too many probable transformations to allow for the insertion of a separate watermark for each transformation, the above described method of compensating for transformations is not optimal. Thus, the present invention includes

an additional improvement, which compensates for arbitrary translation of the image between the times of watermark insertion and watermark detection.

[0093] Arbitrary translation is compensated for in two ways. One way is by translations by even multiples of 8 pixels in the x or y directions when 8×8 blocks are used. This can be easily compensated for if the following restrictions are imposed on the relationship between an index of function h_m(i,j), m, and the row and column numbers of blocks, r and c. We determine the index m according to

$$m = f(r,c) \bmod M,$$

where f(r,c) is a linear function of r and c, and f(0,1) and f(1,0) are integers.

[0094] In addition, let h_m(i,j) be a function expressed as

$$h\_m(i,j) = (h\_0(i,j) + a \times m) \bmod N,$$

where a is an integer. These restrictions are assumed in the remaining part of the detailed description. In this case, the shift compensation is performed in the basic detection algorithm by computing the correlation of the extracted watermark with all cyclical shifts of the watermark being tested for. This is because the values VB_m[i,j] (m=0,...,M-1) obtained from a watermarked image shifted by a multiple of 8 pixels in the horizontal and/or the vertical directions are identical to the values VB_m[i,j] (m=0,...,M-1) obtained from the non-shifted image except that the indices m are cyclically shifted. As a result, the extracted value V[1,...,N] obtained from a watermarked image shifted by a multiple of 8 are identical to the value V[1,...,N] obtained from the non-shifted image except that the elements V[k] are cyclically shifted. For n×n grid format, the same thing is true of a shift of a multiple of n in the horizontal and/or the vertical directions.

[0095] For shifts of less than 8 pixels in x and/or y directions, an exhaustive search can be performed of all 64 possibilities and the maximum Z value taken from the set of 64×M - Z values, where M is the number of 8×8 accumulators. In our tests M was chosen to be 64. The factor M is necessary to account for the cyclic shifts that are introduced by shifts of a multiple of 8 pixels.

[0096] The exhaustive search requires shifting the M 8×8 accumulator array in the spatial domain and then performing the DCT of each of the M 8×8 blocks. This is performed 64 times, once for each of the possible shifts. Thus it is necessary to perform 64×M 8×8 DCTs. If this computation is too expensive in terms of time or memory an alternative method can be used, as described below.

[0097] The second way, which compensates for translations of non-even multiples of n pixels, uses a pattern (referred to as a "registration pattern") which can be inserted at the time of watermark insertion. By finding the location where the registration signal best matches a predefined signal, a detector can determine how much to shift the data before extracting the watermark. This shifting must be done in the spatial domain, but can be done with accumulators, so conversions of whole images are avoided.

[0098] Moreover, the 64×M 8×8 DCTs are unnecessary. Instead, the correct registration is determined in the spatial domain and then compensated for by shifting the pixels in the accumulator arrays. The M 8×8 accumulators are only then transformed into the DCT domain and the watermark extraction is performed as described above.

[0099] The registration pattern is an 8×8 spatial pattern inserted into the image in such a way that the sum of all 8×8 pixel blocks highly correlates with the pattern. Again, an n×n spatial pattern is used when the video is nxn blocks. A registration pattern can be inserted by using the watermark insertion method described above. The sum of all 8×8 pixel blocks becomes highly correlated with a registration pattern if the DCT coefficients of the sum block and those of the registration pattern are highly correlated to each other. In addition, the DCT coefficients of the sum block equals the sum of all DCT blocks because of the linearity of DCT. We can thus insert a registration pattern using a method similar to that inserting a watermark described above, considering the DCT coefficients of a registration pattern as a watermark W and considering the sum of all 8×8 DCT blocks as a value V.

[0100] Insertion is performed by converting the registration pattern into the DCT domain, and then using the basic insertion algorithm with different DCT coefficients D_ks and filter coefficients F_ks. Specifically, each AC term of the registration pattern's DCT is considered one element of the watermark, W[k]. The set of DCT terms that are summed together to extract this element, D_k, is simply the set of corresponding terms of all the 8×8 DCTs in the image. All the F_ks are set to 1. Using these D_ks and F_ks, the insertion algorithm inserts a registration pattern along with each watermark. The watermarks are still inserted with the original D_ks and F_ks.

[0101] During detection, registration is performed as follows. Each of a predetermined number of blocks is summed together to form a single n×n, typically 8×8, block. We have arbitrarily used 64 blocks in our tests. This block contains a registration pattern placed there by the insertion process. To determine the horizontal and vertical translation of the frame, a correlation process is performed in the spatial domain to determine these offsets. Sixty-four correlations are performed for each of the 8 horizontal and 8 vertical motions that are possible. When the 8×8 patch is shifted either horizontally or vertically, a wrap around shift is performed.

[0102] We now describe how to determine the shift of the grid. In the following method, we assume the integer values f(0,1) and f(1,0) are relatively prime to M.

[0103] First, the blocks for which the same function h_m(i,j) is used in classifying DCT coefficients are added together in the spatial domain for generating a summed block VB_m[i,j] for each m. The sum of all n×n blocks denoted by AB[i,j], is computed by

$$AB[i,j] = \sum_{m=0}^{M-1} VB\_m[i,j] \qquad (i=1,...,n,\ j=1,...,n).$$

[0104] Then the correlation coefficient between AB[i,j] and a registration pattern, denoted by R[i,j], is computed. After calculation of the correlation coefficient, the values AB[i,j] are cyclically shifted by one column in the horizontal direction, and the correlation coefficient between AB[i,j] and R[i,j] is calculated in the same way. The same operations are repeated for each shift. After shifting n times, AB[i,j] becomes identical to AB[i,j] before any shift is done. Then, the values AB[i,j] are cyclically shifted by one row in the vertical direction, and calculation of a correlation coefficient and shift by one column in the horizontal direction are repeated. In this way, we can calculate correlation coefficients for all n^2 possible shifts. At the same time, we search the shift value (offset), denoted by (X,Y), which gives the maximum correlation coefficient.

[0105] After the-offset (X,Y) has been determined, the M summed blocks VB_m[i,j] are then shifted accordingly in the spatial domain.

[0106] Next, we describe the method to compensate for the shift value X in the horizontal direction. To do so, the values VB_m[i,j] (m=0,...,M-1) are first copied on an n×nM array VB1[i,j] (i=1,...,n, j=1,...,nM). In copying the values VB_m[i,j], the spatial relationships between blocks, that is, which blocks are adjoining a certain block are considered. The function f(r,c) is linear, so

$$f(r,c+1) = f(r,c) + f(0,1).$$

[0107] This indicates that the blocks whose index m equals

$$m1 = (m0 + f(0,1)) \bmod M$$

are located next to the blocks in which m=m0. So, the values VB_m1 are copied next to the values VB_m0 in the array VB1. For this reason, for each I (I=0,...,M-1), the values VB_m'[i,j] (i=1,...,n, j=1,...,n, m'=I×f(0,1) mod M) are copied in the n×n square region of VB1 where the index of the left-top corner is (1, nI+1). After copying the values for all I, the array VB1 is filled with the values VB_m[i,j] because f(0,1) and M are relatively prime. Next, the values in the array VB1 are cyclically shifted by X in the horizontal direction. Then the values in VB1 are returned to VB_m in such a way that the value in the n×n region of VB1 where the index of the left-top corner is (1, nI+1) are substituted to VB_m'[i,j] (i=1,...,n, j=1,...,n, m'=I×f(0,1) mod M) for each I (I=0,...,M-1). The horizontal offset value X can thus be compensated for without shifting the whole image itself.

[0108] Next, the shift value Y in the vertical direction is compensated for. To do so, the values VB_m[i,j] (m=0,...,M-1) are first copied on an nM×n array VB2[i,j] (i=1,...,nM, j=1,...,n). In copying the values VB_m[i,j], the spatial relationship between blocks are considered. The function f(r,c) is linear, so

$$f(r+1,c) = f(r, c) + f(1,0).$$

[0109] This indicates that the blocks whose index m equals

$$m1 = (m0 + f(1,0)) \bmod M$$

locate under the blocks in which m=m0. So, the values VB_m1 are copied under the values VB_m0 in the array VB2. For this reason, for each I (I=0,...,M-1), the values VB_m'[i,j] (i=1,...,n, j=1,...,n, m'=I×f(0,1) mod M) are copied on the n×n square region of VB2 where the index of the left-top corner is (nI+1, 1). After copying the values for all I, the array VB2 is filled with the values VB_m[i,j] because f(1,0) and M are relatively prime. Next, the values in the array VB2 are cyclically shifted by Y in the vertical direction. Then the values in VB2 are returned to VB_m in such a way that the value in the n×n region of VB2 where the index of the left-top corner is (nI+1, 1) are substituted in VB_m'[i,j] (i=1,...n, j=1,...,n, m'=I×f(0,1) mod M) for each I (I=0,...,M-1). The vertical offset value Y can thus be compensated without shifting the whole image itself.

[0110] The offset of the n×n grid is compensated by the process mentioned above. These processes are performed in the registration process 108 (Figure 10) as will be explained later. A shift of a multiple of n remains even after these processes, but this shift does not affect the watermark detection because the correlation coefficient between a watermark W and an extracted value V is calculated by shifting the watermark W cyclically as described above. After the registration process above is applied, the M blocks VB_m (m=0,...,M-1) are transformed back to the DCT domain.

[0111] With reference now to Figures 10 and 11, there are shown the basic detection algorithms modified to compensate for translational registration. In the case of MPEG video input (Figure 10), 8×8 DCT blocks obtained from an MPEG video stream are first classified into M groups according to their indices m of the function h_m(i,j), summed within the groups for generating M summed blocks, and the resultant summed blocks are accumulated in 8×8 accumulators 102. The M summed blocks in accumulators 102 must be converted into the spatial domain by performing an inverse DCT operation in inverse DCT converter 104, and accumulated in accumulators 106. Finding the offset value of the 8×8 grid and compensating for the offset is executed for the output from 8×8 accumulators 106 in registration 108 as described above. The registration data outputted from registration process 108 is accumulated in accumulators 110 and converted into the DCT domain in DCT converter 112 for watermark extraction by use of accumulators 114, watermark extractor 116 and watermark decoder 118. In watermark extractor 116, the DCT coefficients outputted from accumulator 114 are classified into N sets according to the functions h_m(i,j) and summed for extracting a watermark. The obtained watermark is provided to watermark decoder 118, in which the processes executed in comparator 84 in Figure 8 for finding a watermark corresponding to the extracted watermark. The watermark considered to have been inserted is outputted from the watermark decoder 118. In the case of uncompressed input data (Figure 11), the input data is divided into 8×8 blocks and accumulated in accumulators 106 according to the indices of the functions h_m(i,j), and registration 108 is performed before conversion into the DCT domain in DCT converter 112. The process continues as described above.

[0112] While there has been described and illustrated methods of insertion and detection of watermarks in image data, it will be understood by those skilled in the art that variations and modifications are possible without deviating from the spirit and broad teachings of the present invention which shall be limited solely by the scope of the claims appended hereto.

Claims

1. A method of inserting a watermark signal into data to be watermarked comprising the steps of:

   applying a watermark extraction to unwatermarked data for generating extracted values;
   computing the difference between a watermark to be inserted and the extracted values; and
   adding the computed difference throughout the unwatermarked data.

2. A method of inserting a watermarked signal into data to be watermarked as set forth in claim 1, wherein said adding the computed difference comprises distributing the computed difference according to a perceptual model of human visual sensitivity.

3. A method of inserting a watermarked signal into data to be watermarked as set forth in claim 1, wherein said adding the computed difference comprises distributing the computed difference according to how affected the data will be to a predetermined attack so that the attack will not affect the watermarked data.

4. A method of inserting a watermarked signal into data to be watermarked as set forth in claim 1, wherein said adding the computed difference comprises distributing the computed difference according to a stochastic method to reduce the susceptibility to tampering.

5. A method of inserting a watermarked signal into data to be watermarked as set forth in claim 1, wherein said unwatermarked data is compressed video and said compressed video is divided into groups of blocks, where said computing the difference computes the difference between a watermark and the sum of the extracted values from each group of blocks and said adding distributes the difference into the group of blocks.

6. A method of inserting a watermarked signal into data to be watermarked as set forth in claim 5, wherein said groups of blocks are selected according to response to predetermined attacks and signal degradations.

7. A method of detecting a watermark from watermarked data comprising the steps of:

   receiving watermarked data where the watermarked data comprises n×n blocks of watermarked data;

summing the n×n blocks of watermarked data to form at least one n×n block of summed watermarked data;
cyclically shifting the block in the spatial domain to register the block; and
extracting the watermark from the at least one n×n block.

8. A method of detecting a watermark from watermarked data comprising the steps of:

receiving watermarked data where the watermarked data comprises n×n blocks of watermarked data;

summing the n×n blocks of watermarked data to form at least one n×n block of summed watermarked data; and

extracting the watermark from the at least one n×n block by summing predetermined watermarked data from each n×n block to yield each element of the extracted watermark.

9. A method of detecting a watermark from watermarked data as set forth in claim 8, wherein said extracting is performed in the spatial domain using at least one n×n blocks.

10. A method of detecting a watermark from watermarked data as set forth in claim 8, wherein said extracting is performed in the frequency domain using at least one n×n blocks.

11. A method of detecting a watermark from watermarked data as set forth in any of claims 8 to 10, wherein each n×n block is filtered before said extracting.

12. A method of detecting a watermark from watermarked data comprising the steps of:

receiving watermarked data where the watermarked data comprises n×n blocks of watermarked data;
summing the n×n blocks of watermarked data to form M n×n blocks of summed watermarked data, where M is the number of functions mapping frequency indices onto the indices of elements of watermarks; and
extracting the watermark from the at least one n×n block.

13. A method of detecting a watermark from watermarked data as set forth in claim 12, wherein said forming is performed in the spatial domain.

14. A method of detecting a watermark from watermarked data as set forth in claim 12, wherein said forming is performed in the frequency domain.

15. A method of inserting a watermark signal into video images comprising the steps of:

receiving input video images;
performing discrete cosine transformation (DCT) of said input video images to obtain input video image DCT values;
adding watermark signals to said input video image DCT values to obtain watermarked DCT values;
performing an inverse DCT on the watermarked DCT values for generating watermarked images;
subtracting predicted images stored in memory from said watermarked images for generating residual images;
performing discrete cosine transformation of said residual images to obtain residual image DCT values and quantizing the residual image DCT values to derive residual image quantized DCT values;
inverse quantizing the residual image quantized DCT values to obtain residual image de-quantized DCT values;
performing discrete cosine transformation of the predicted images to obtain predicted image DCT values;
summing the residual image de-quantized DCT values and the predicted image DCT values for generating decoded image DCT coefficients;
calculating correction signals by applying a watermark extraction to the decoded image DCT coefficients and comparing resultant extracted values with watermark target values;
adding the correction signals to the residual image quantized DCT values for obtaining an output signal;
inverse quantizing the output signal to obtain inverse quantized output data, perforing inverse DCT of the inverse quantized output signal and summing the resultant signal with the predicted images for generating a summed signal;
generating predicted images to be stored in said memory by using the summed signal; and

variable length encoding the output signal for providing a watermarked MPEG video signal of the input video images.

16. A method of inserting a watermark signal into video images comprising the steps of:

receiving input video images;
performing discrete cosine transformation (DCT) of said input video images to obtain input video image DCT values;
adding watermark signals to said input video image DCT values to obtain watermarked DCT values;
performing discrete cosine transformation on predicted images stored in memory to obtain predicted image DCT values;
subtracting the predicted image DCT values from the watermarked DCT values to obtain residual image DCT values;
quantizing the residual image DCT values to obtain residual image quantized DCT values;
inverse quantizing the residual image quantized DCT values to obtain residual image de-quantized DCT values and summing the residual image de-quantized DCT values with the predicted image DCT values, for generating decoded image DCT coefficients;
calculating correction signals by applying a watermark extraction to the decoded image DCT coefficients and comparing resultant extracted values with watermark target values;
summing the correction signals with the residual image quantized DCT values to obtain an output signal;
inverse quantizing the output signal to obtain inverse quantized output signal, performing inverse DCT of the inverse quantized output signal and summing the resultant signal with the predicted images to provide locally-decoded images;
generating predicted images to be stored in said memory using the locally-decoded images; and
valuable length encoding the output signal to provide a watermarked MPEG video signal of the input video images.

17. A method of inserting a watermark signal into video images as set forth in claim 15 or 16, wherein said calculating step generates a negative value as said correction signal when the extracted value exceeds the corresponding value of said watermark target values, and generates a positive value as said correction signal when the extracted value falls short of the corresponding value of said watermark target values.

18. A method of inserting a watermark signal into video images as set forth in claim 17, wherein said calculating step generates a zero value as the correction signal if the corresponding value of said residual image quantized DCT values is zero.

19. A method of inserting a watermark signal into video images as set forth in claim 17 or 18, wherein said positive value is +1 and said negative value is -1.

20. A method of inserting an extractable watermark signal into image data so that the watermark signal can be extracted after the watermarked data is subject to distortion comprising the steps of:

providing image data to be watermarked;
copying the image data;
applying a predefined distortion to the copy of image data;
inserting a watermark signal into the distorted copy of image data;
subtracting the copy of distorted image data from the watermarked copy of distorted image data to generate a watermark pattern;
performing inverse distortion on the watermark pattern to generate the watermarked pattern for nondistorted image data; and
combining said image data to be watermarked and said watermark pattern for nondistorted image data to obtain watermarked data.

21. A method of inserting a watermark signal into image data as set forth in claim 20, wherein said predetermined distortion is panscan mode image data.

22. A method of inserting a watermark signal into image data as set forth in claim 20, wherein said predetermined distortion is letterbox mode image data.

23. A method of inserting a watermark signal into image data as set forth in claim 20, wherein said combining includes combining with perceptual modeling.

24. A method of inserting an extractable watermark signal into image data so that the watermark signal can be extracted after the watermarked data is subject to distortion comprising the steps of:

receiving image data in form of n×n blocks; and

inserting an n×n registration pattern into the image data.

25. A method of inserting an extractable watermark signal into image data as set forth in claim 24, wherein said inserting is performed by converting the registration pattern into the DCT domain and then inserting the DCT values of the registration pattern into the image data.

26. A method of detecting a watermark from watermarked image data that has been subjected to a two dimensional translation distortion comprising the steps of:

partitioning received distorted watermarked image data into a set of n×n blocks;
summing together watermarked image data in a predetermined number of n×n blocks;
if summed n×n blocks are not in spatial domain, transforming summed n×n blocks into spatial domain;
determining two-dimensional translation distortion of the summed n×n blocks;
shifting the summed n×n blocks according to the determined distortion;
transforming the shifted n×n blocks into transform domain;
extracting a possible watermark signal from the transformed n×n blocks;
correlating the possible watermark with a known watermark;
repeating said correlating for cyclic shifts of the possible watermark; and
determining the presence of a watermark based on said correlating.

27. A method of detecting a watermark from watermarked image data as set forth in claim 26, where the two dimensional translation is a geometric shift in at least one of the horizontal or vertical direction.

28. A method of detecting a watermark from watermarked image data that has been subjected to a two dimensional translation distortion comprising the steps of:

(a) partitioning received distorted watermarked image data in n×n blocks;
(b) summing together watermarked image data in a predetermined number of n×n blocks;
(c) if summed n×n blocks are not in spatial domain, transforming summed n×n blocks into spatial domain;
(d) shifting the summed n×n blocks by one of n^2 possible shifts;
(e) transforming the shifted n×n blocks into transform domain;
(f) extracting a possible watermark signal from the transformed n×n blocks;
(g) correlating the possible watermark with a known watermark;
(h) repeating said correlating for cyclic shifts of the possible watermark;
(i) repeating steps (d)-(h) for each of the n^2 possible shifts; and
(j) determining the presence of a watermark based on the results of step (i).

29. A method of detecting a watermark from watermarked image data as set forth in claim 26 or 28, wherein said n×n block is a 8×8 block.

30. A method of detecting a watermark from watermarked image data as set forth in claim 26 or 28, wherein said correlating is performed for n^2 possible shifts.

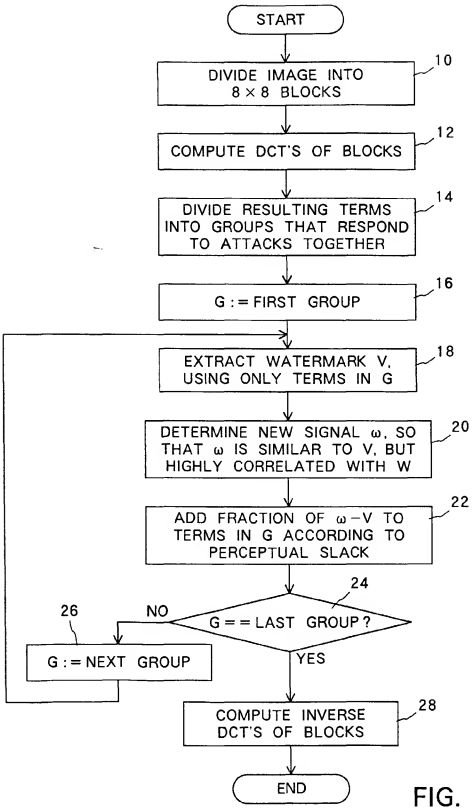
START
DIVIDE IMAGE INTO 8 × 8 BLOCKS
10
COMPUTE DCT'S OF BLOCKS
12
DIVIDE RESULTING TERMS INTO GROUPS THAT RESPOND TO ATTACKS TOGETHER
14
G := FIRST GROUP
16
EXTRACT WATERMARK V, USING ONLY TERMS IN G
18
DETERMINE NEW SIGNAL ω, SO THAT ω IS SIMILAR TO V, BUT HIGHLY CORRELATED WITH W
20
ADD FRACTION OF ω−V TO TERMS IN G ACCORDING TO PERCEPTUAL SLACK
22
24
G = = LAST GROUP?
NO
YES
26
G := NEXT GROUP
COMPUTE INVERSE DCT'S OF BLOCKS
28
END

FIG. 1

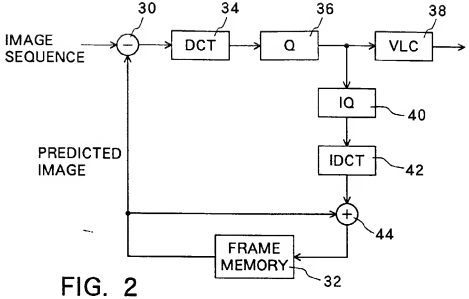
30
34
36
38
IMAGE SEQUENCE
DCT
Q
VLC
IQ
40
PREDICTED IMAGE
IDCT
42
44
FRAME MEMORY
32

FIG. 2

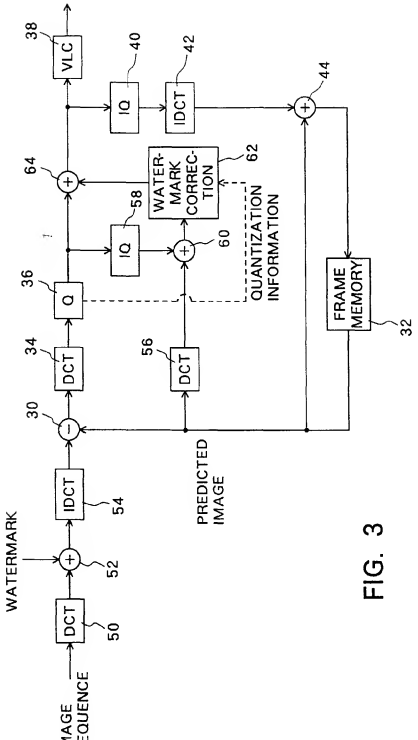
IMAGE SEQUENCE
DCT
50
WATERMARK
52
IDCT
54
PREDICTED IMAGE
30
DCT
34
Q
36
DCT
56
IQ
58
60
WATER-MARK CORREC-TION
62
64
QUANTIZATION INFORMATION
IQ
40
IDCT
42
VLC
38
44
FRAME MEMORY
32

FIG. 3

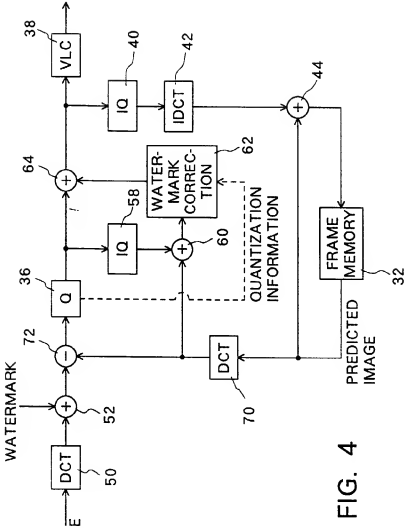
IMAGE SEQUENCE
WATERMARK
DCT
50
52
72
Q
36
IQ
58
60
64
WATER-MARK CORREC-TION
62
QUANTIZATION INFORMATION
VLC
38
IQ
40
IDCT
42
44
FRAME MEMORY
32
PREDICTED IMAGE
DCT
70

FIG. 4

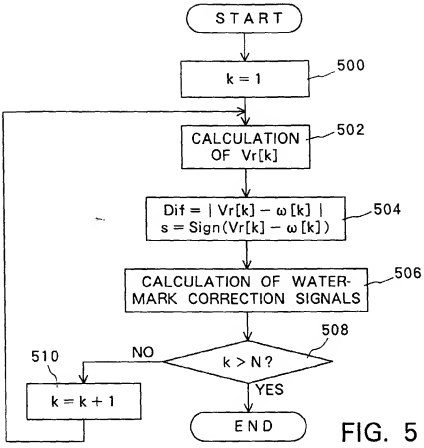
START
k = 1
500
CALCULATION OF Vr[k]
502
Dif = | Vr[k] − ω[k] |
s = Sign(Vr[k] − ω[k])
504
CALCULATION OF WATER-MARK CORRECTION SIGNALS
506
508
k > N?
NO
YES
510
k = k + 1
END

FIG. 5

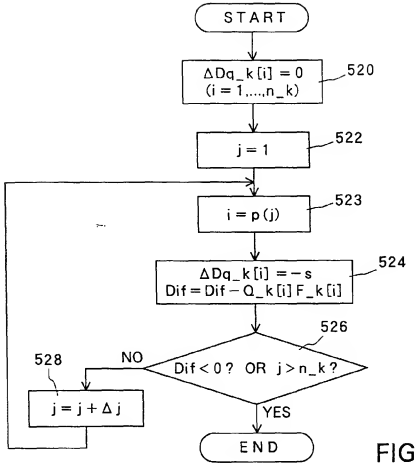
START
ΔDq_k[i] = 0
(i = 1,...,n_k)
520
j = 1
522
i = p(j)
523
ΔDq_k[i] = −s
Dif = Dif − Q_k[i] F_k[i]
524
526
Dif < 0 ? OR j > n_k ?
NO
528
j = j + Δj
YES
END

FIG. 6

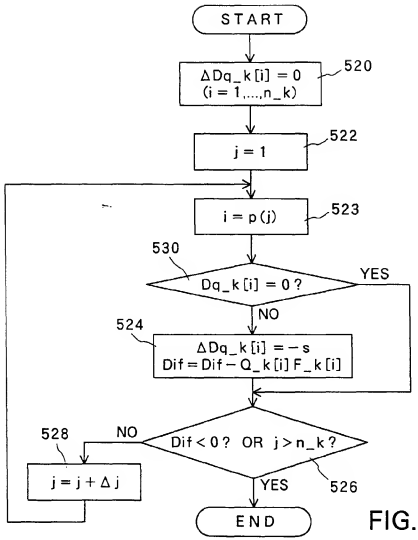
START
ΔDq_k[i] = 0
(i = 1,...,n_k)
520
j = 1
522
i = p(j)
523
530
Dq_k[i] = 0 ?
YES
NO
524
ΔDq_k[i] = −s
Dif = Dif − Q_k[i] F_k[i]
528
NO
Dif < 0 ? OR j > n_k ?
526
YES
j = j + Δj
END

FIG. 7

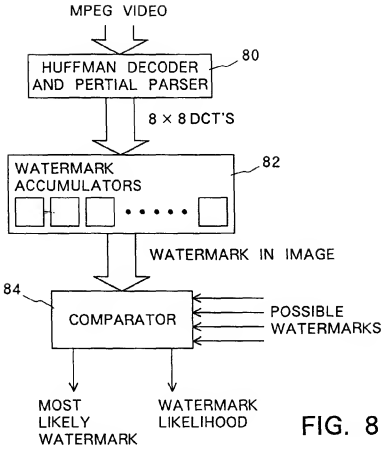
MPEG VIDEO
HUFFMAN DECODER AND PERTIAL PARSER
80
8 × 8 DCT'S
WATERMARK ACCUMULATORS
82
WATERMARK IN IMAGE
84
COMPARATOR
POSSIBLE WATERMARKS
MOST LIKELY WATERMARK
WATERMARK LIKELIHOOD

FIG. 8

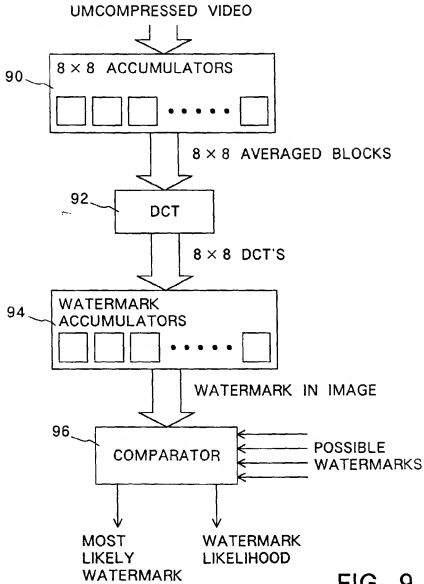
UMCOMPRESSED VIDEO
8 × 8 ACCUMULATORS
90
8 × 8 AVERAGED BLOCKS
92
DCT
8 × 8 DCT'S
WATERMARK
ACCUMULATORS
94
WATERMARK IN IMAGE
96
COMPARATOR
POSSIBLE
WATERMARKS
MOST
LIKELY
WATERMARK
WATERMARK
LIKELIHOOD

FIG. 9

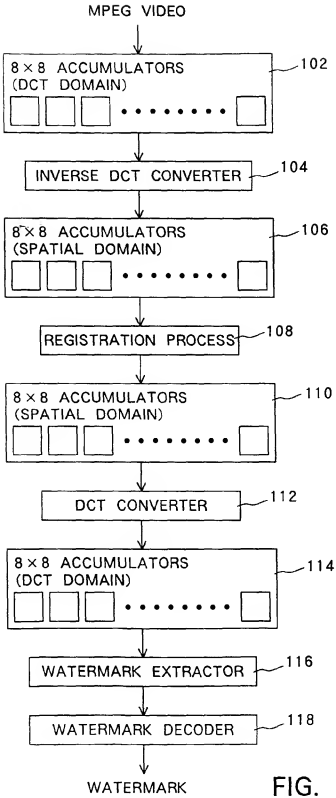
MPEG VIDEO
8 × 8 ACCUMULATORS (DCT DOMAIN)
102
INVERSE DCT CONVERTER
104
8 × 8 ACCUMULATORS (SPATIAL DOMAIN)
106
REGISTRATION PROCESS
108
8 × 8 ACCUMULATORS (SPATIAL DOMAIN)
110
DCT CONVERTER
112
8 × 8 ACCUMULATORS (DCT DOMAIN)
114
WATERMARK EXTRACTOR
116
WATERMARK DECODER
118
WATERMARK

FIG. 10

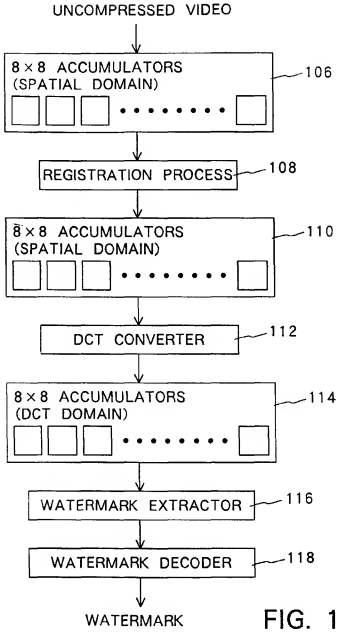
UNCOMPRESSED VIDEO
8×8 ACCUMULATORS (SPATIAL DOMAIN)
106
REGISTRATION PROCESS
108
8×8 ACCUMULATORS (SPATIAL DOMAIN)
110
DCT CONVERTER
112
8×8 ACCUMULATORS (DCT DOMAIN)
114
WATERMARK EXTRACTOR
116
WATERMARK DECODER
118
WATERMARK

FIG. 11